FORSCHUNGSZENTRUM JÜLICH GmbH
Zentralinstitut für Angewandte Mathematik
D-52425 Jülich, Tel. (02461) 61-6402

Interner Bericht

Höchstleistungsrechenzentrum: Verteilter massiv-paralleler Rechner

Jörg Henrichs, Wolfgang E. Nagel, Michael Weber,
Roland Völpel, Helmut Grund**

FZJ-ZAM-IB-9711

Juli 1997

(letzte Änderung: 08.07.97)

(*) Institut für Algorithmen und Wissenschaftliches Rechnen
GMD - Forschungszentrum Informationstechnik GmbH, Schloß Birlinghoven

Erscheint als Abschlußbericht des Teilprojektes „Höchstleistungsrechenzentrum: Verteilter massiv-paralleler Rechner“ im „Regionalen Testbed Nordrhein-Westfalen“ (RTB-NRW) beim DFN-Verein.

Höchstleistungsrechenzentrum: Verteilter massiv-paralleler Rechner

J. Henrichs[†], W.E. Nagel[†], M. Weber[†],
R. Völpel[‡], H. Grund[‡]

[†] Zentralinstitut für Angewandte Mathematik
Forschungszentrum Jülich GmbH
email: {j.henrichs,w.nagel}@fz-juelich.de

[‡] Institut für Algorithmen und Wissenschaftliches Rechnen
GMD - Forschungszentrum Informationstechnik GmbH, Schloß Birlinghoven
email: {voelpel,grund}@gmd.de

1 Einleitung

In den vergangenen Jahren hat sich die Idee, eine Anwendung durch ein System unterschiedlicher Rechner - einen sogenannten *Metacomputer* - gemeinsam bearbeiten zu lassen, konkretisiert. So lassen sich z.B. bei der Computersimulation natürlicher Vorgänge bereits auf der Ebene der Modelle funktionale Unterscheidungen treffen, die in vielen Fällen unzureichend von einem einzigen System abgedeckt werden können. Bei Problemen aus der *Computational Fluid Dynamics (CFD)* genügt oft nicht nur die Betrachtung des Flusses, eine realistische Simulation muß auch die chemischen Reaktionen der beteiligten Partikel berücksichtigen. Ähnlich ist die Situation bei Klimamodellen, in denen Berechnungen der Atmosphäre an Ozeansimulationen gekoppelt werden. Gegenwärtig werden derartige Rechnungen aus dem Bereich der sogenannten *Coupled Fields* [7] von getrennten Programmen durchgeführt, und ein Programm liest die Zwischenresultate des anderen als Eingabedatei.

Ein wichtigerer und vielversprechender Ansatz ergibt sich durch die fortgesetzte Entwicklung unterschiedlicher Rechner-Architekturen wie z.B. Superskalar-, Vektor- oder Parallelrechner. Rechenintensive Anwendungen besitzen typischerweise aber Programmteile mit unterschiedlichen Hardware-Anforderungen. In der Regel bestimmt hier der zeitlich dominierende Teil das Zielsystem. Ein Metacomputer, der hinter seiner einheitlichen Systemsicht unterschiedliche Rechnertypen vereint, bietet insbesondere die Möglichkeit, auf Anforderungen flexibel zu reagieren und die jeweils geeignete Architektur für einzelne Programmteile zu nutzen. Durch die sich somit ergebende akkumulierte Leistung ergibt sich auch der rein

ökonomische Vorteil, daß sich durch die Nutzung eines Metacomputers die Neuanschaffung von Rechnern erübrigen kann.

Ein Bedarf für das Metacomputing ist also offensichtlich, wenngleich Realisierungen bislang noch nicht über das Prototypstadium hinaus sind. Ziel des Teilprojekts Z2 "Höchstleistungsrechenzentrum: Verteilter massiv-paralleler Rechner" im "Regionalen Testbeds NRW" (RTB NRW) war der Pilotbetrieb einer massivparallelen Anwendung über ein Hochleistungs-Datennetz. Dazu mußte zuerst eine einheitliche Schnittstelle gefunden werden, mittels derer der Datenaustausch zwischen den Rechnern realisiert werden kann, die aber auch das breitbandige Netz nutzt. Ein weiterer wichtiger Punkt war der Test von Werkzeugen, die die Programmierung in einer heterogenen Umgebung unterstützen können.

Im weiteren wird zuerst das RTB-NRW vorgestellt. Insbesondere werden hier Meßergebnisse gezeigt, die den Unterschied zwischen theoretischer und tatsächlich meßbarer Leistung aufzeigen. Danach wird auf die benutzten Rechner eingegangen. Erwähnenswert ist hier insbesondere die Tatsache, daß die ursprünglich für die Nutzung durch das Projekt geplante TMC CM-5 während der Projektlaufzeit abgeschafft und in Vertretung die IBM-SP2 der GMD in das Projekt eingebunden werden mußte. Anschließend wird auf die besondere Problematik der Parallelisierung für ein heterogenes System anhand des Programmes TRACE eingegangen. Hierbei werden insbesondere wichtige Software-Produkte, wie Kommunikations-Bibliotheken und Visualisierungs-Tools vorgestellt. Nach den Meßergebnissen wird eine weitere Anwendung beschrieben, die ebenfalls auf Eignung für ein heterogenes System untersucht wurde. Der Ausblick faßt schließlich die Erkenntnisse zusammen, die aus diesem Projekt gewonnen wurden.

2 Das Regionale Testbed NRW

Die Voraussetzung zur effizienten Nutzung zweier Rechner bei der gemeinsamen Bearbeitung eines Programms ist die Existenz eines leistungsfähigen Kommunikationsnetzes. Während in der Vergangenheit derartige Bedingungen allenfalls innerhalb eines Campus oder Rechenzentrums vorlagen, konnte im Rahmen des RTB-NRW eine ATM-Verbindung zwischen den Standorten Aachen, Bonn, Köln, Jülich und St. Augustin realisiert werden. Leider konnte die in der Phase der Feinspezifikation vorausgesetzte Bandbreite von 155 Mbit/s aus finanziellen Gründen nicht bereitgestellt werden, es stand letztlich nur eine Verbindung mit 34 Mbit/s zur Verfügung. Diese Verbindung wurde Mitte März 1995, sechs Monate nach Projektbeginn, geschaltet. Abbildung 1 zeigt die Netzwerktopologie des RTB-NRW [10]. Eine für das Metacomputing wichtige ATM-Eigenschaft, die in den bisherigen LAN- und WAN-Kommunikationstechnologien in dieser Form nicht zur Verfügung steht, ist die Option *quality of service*, die es erlaubt, für eine Anwendung einen festen Datendurchsatz zu reservieren. Störungen durch weitere Netzbenutzer sind aufgrund der Bandbreitenreservierung dann nicht mehr

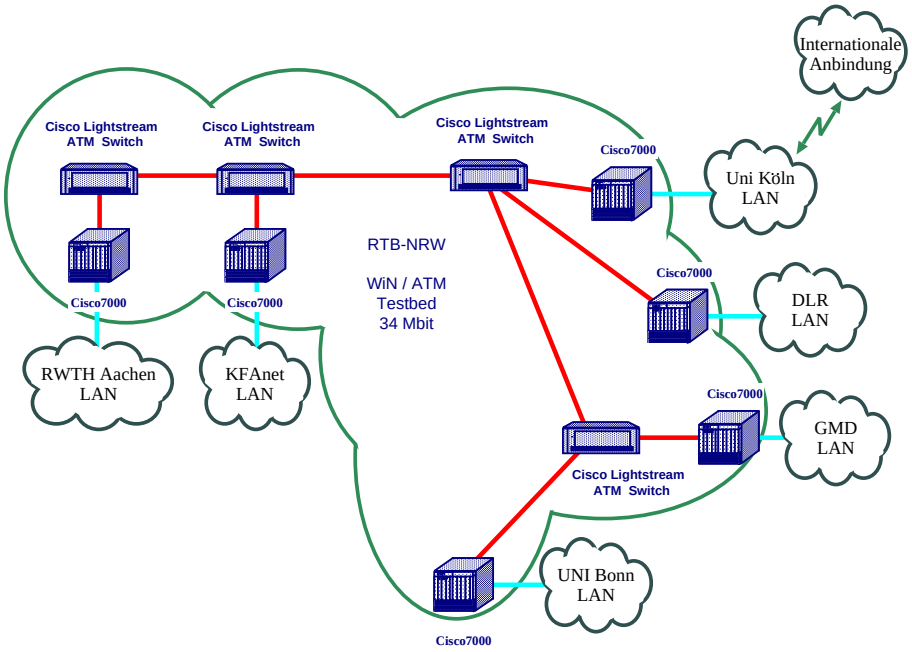

Abbildung 1: Die Netzwerktopologie des RTB-NRW.

möglich, die angeforderte Bandbreite kann voll von der Anwendung genutzt werden. Da die verwendete Infrastruktur heute noch nicht über eine *native ATM*-Schnittstelle verfügt, mußte weiterhin *classical IP* über ATM genutzt werden. Insbesondere steht die Option *qualitiy of service* somit derzeit noch nicht zur Verfügung. Da die Leitung zwischen der GMD und der KFA auch von der RWTH Aachen und der Universität zu Köln benutzt wurde, stand keine dedizierte Verbindung zur Verfügung. Statistiken haben gezeigt, daß derzeit auf der Verbindung zwischen KFA und Universität zu Köln tagsüber ca. 4 Mbit/s durchschnittlich benutzt wurden. Dementsprechend waren die Meßergebnisse teilweise von starken Schwankungen geprägt, so daß zahlreiche Messungen gemacht werden mußten, um einen zuverlässigen Mittelwert zu erhalten.

Von Bedeutung erwies sich erwartungsgemäß auch die Art der Anbindung der jeweiligen Rechner an das RTB-NRW, da die tatsächlich meßbare Leistung zwischen den Rechnern auch von der Anbindung der jeweiligen Rechner innerhalb der lokalen Netze abhängt:

- Der Zugang zum 34 Mbit ATM-Netz im RTB-Verbund konnte in der GMD theoretisch ab März 1995 über FDDI realisiert werden, allerdings verhinderten Hardware-Probleme der SP2 einen rechtzeitigen Beginn. Darüber hinaus stand in der Folge auch ein direkter Zugang zum ATM-Testbed zur Verfügung, so daß eine direkte ATM-Verbindung (mittels *classical IP*) zwischen der seit November 1995 mit 2 ATM-Schnittstellen ausgerüsteten GMD SP2 und externen Systemen möglich wurde.

- Für die Paragon der KFA gab und gibt es leider keine ATM-Karte, da die Firma die Fertigung dieser Rechnerlinie eingestellt hat und deshalb die Entwicklung einer Karte für dieses Marktsegment wirtschaftlich nicht interessant ist. Deshalb erfolgte die Anbindung mittels eines NSC-Routers mit einer HiPPI-Verbindung zur Intel Paragon auf der einen und einem FDDI-Anschluß an das KFA-interne LAN auf der anderen Seite. Das KFA-LAN wurde dann über einen Router Cisco 7000 und einen Switch Cisco Lightstream 100 mit dem RTB-NRW verbunden (vgl. Abb. 3).

Durch diese alles andere als optimale Anbindung der Maschinen an das Netz blieb auch die bei den ersten Tests gemessene Performance deutlich hinter den Erwartungen zurück. Bereits bei der Analyse des lokalen Verkehrs fiel auf, daß die Paragon eine vergleichsweise schlechte Netzanbindung über TCP/IP zeigt. Zur Klärung wurden umfangreiche Messungen sowohl innerhalb der jeweiligen LANs als auch über das RTB durchgeführt. Um Einflüsse durch zusätzliche Softwareschichten zu vermeiden, wurden die Messungen auf der Ebene der UNIX-*sockets* aufgesetzt. Dazu wurde ein Pingpong-Programm der Art verwendet, wie es typischerweise auch zur Bestimmung der Kommunikationsleistung interner Netzwerke von Parallelrechnern üblich ist [10]. Tabelle 1 faßt diese Ergebnisse zusammen.

Performance-Matrix

– Socket-Kommunikation –

		KFA					GMD	
		WS (FDDI)	SP2 (FDDI)	WS (ATM)	XP/S (Eth.)	XP/S (FDDI)	SP2 (Eth.)	SP2 (ATM)
K:WS (FDDI)	A:		0.7	1		5		
	B:		5200	4600		680		
K:SP2 (FDDI)	A:	0.7	0.7					
	B:	5200	5200					
K:WS (ATM)	A:	1						3.6/4.5
	B:	4600						2700/1000
K:XP/S (Eth.)	A:				7		8	
	B:				320		200	
K:XP/S (FDDI)	A:	5						
	B:	680						
G:SP2 (Eth.)	A:		5		8			
	B:		450		200			
G:SP2 (ATM)	A:			3.6/4.5				0.6
	B:			2700/1000				9200

Tabelle 1: Latenz (A:)[ms] und Bandbreite (B:) [kB/s]

Zuerst wurde die KFA-interne FDDI-Leistung gemessen. Dabei ergab sich sowohl von einer *workstation* als auch von der IBM-SP2 der KFA eine Bandbreite von ca. 5.2 MByte/s bei einer Latenz von 0.7 ms. Bei einer mit ATM angeschlossenen *workstation* ergaben sich die etwas geringeren Werte von 4.6 MByte/s Bandbreite und eine Latenz von 1 ms, erwartungsgemäß etwas schlechtere Werte, da ein

zusätzlicher *switch* notwendig ist. Überraschend ist allerdings, daß sich bei der Paragon XP/S 10 nur eine Bandbreite von 0.68 MByte/s bei einer Latenz von 5 ms ergab. Eine direkte Gegenüberstellung in Abb. 2 zeigt die geringe Leistungsfähigkeit der Paragon im Vergleich zu anderen Systemen. Obwohl die Datenpakete von

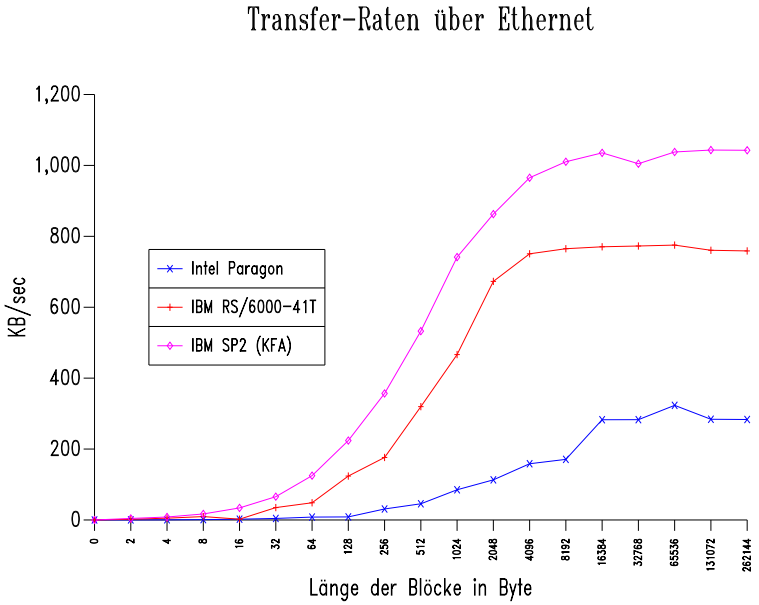

Abbildung 2: IP-Kommunikationsleistung für verschiedene Systeme

und zur Paragon, wie in Abb. 3 zu sehen ist, einen längeren Weg, nämlich zwei lokale Router bzw. FDDI Ringe, durchlaufen, erklärt dies allein noch nicht den deutlichen Leistungsrückgang. Denn auch bei einer TCP/IP-Verbindung innerhalb der Paragon selbst ergibt sich der sehr schlechte Wert von 0.32 MByte/s bei einer Latenz von 7 ms. Andere Paragon-Standorte haben die geringe TCP/IP-Leistungsfähigkeit der Paragon bestätigt. Gespräche mit der Firma Intel ergaben, daß aufgrund der speziellen IP-Implementierung keine Verbesserungen für die Paragon zu erwarten sind.

Bei einer externen Verbindung zwischen der KFA und der GMD zeigt sich ein ähnliches Verhalten: bei einer normalen Ethernet-Verbindung zwischen der GMD-SP2 und der KFA-SP2 ergibt sich eine Bandbreite von 0.45 MByte/s bei einer Latenz von 5 ms. Wird hingegen die Paragon XP/S 10 genutzt, verschlechtern sich die Werte auf eine Bandbreite von 0.2 MByte/s mit 8 ms Latenz. Auch dies ist ein deutliches Indiz auf die schlechte TCP/IP-Implementation der Paragon.

Da jeder Router eine zusätzliche Verzögerung von rund 1 ms hervorruft, wurden Alternativen untersucht, die eine *durchgehende* ATM-Verbindung erlauben und somit eine bessere Performance ermöglichen. Da in der KFA weder die Paragon noch die ebenfalls installierte IBM SP2 mit einem ATM-Interface ausgerüstet

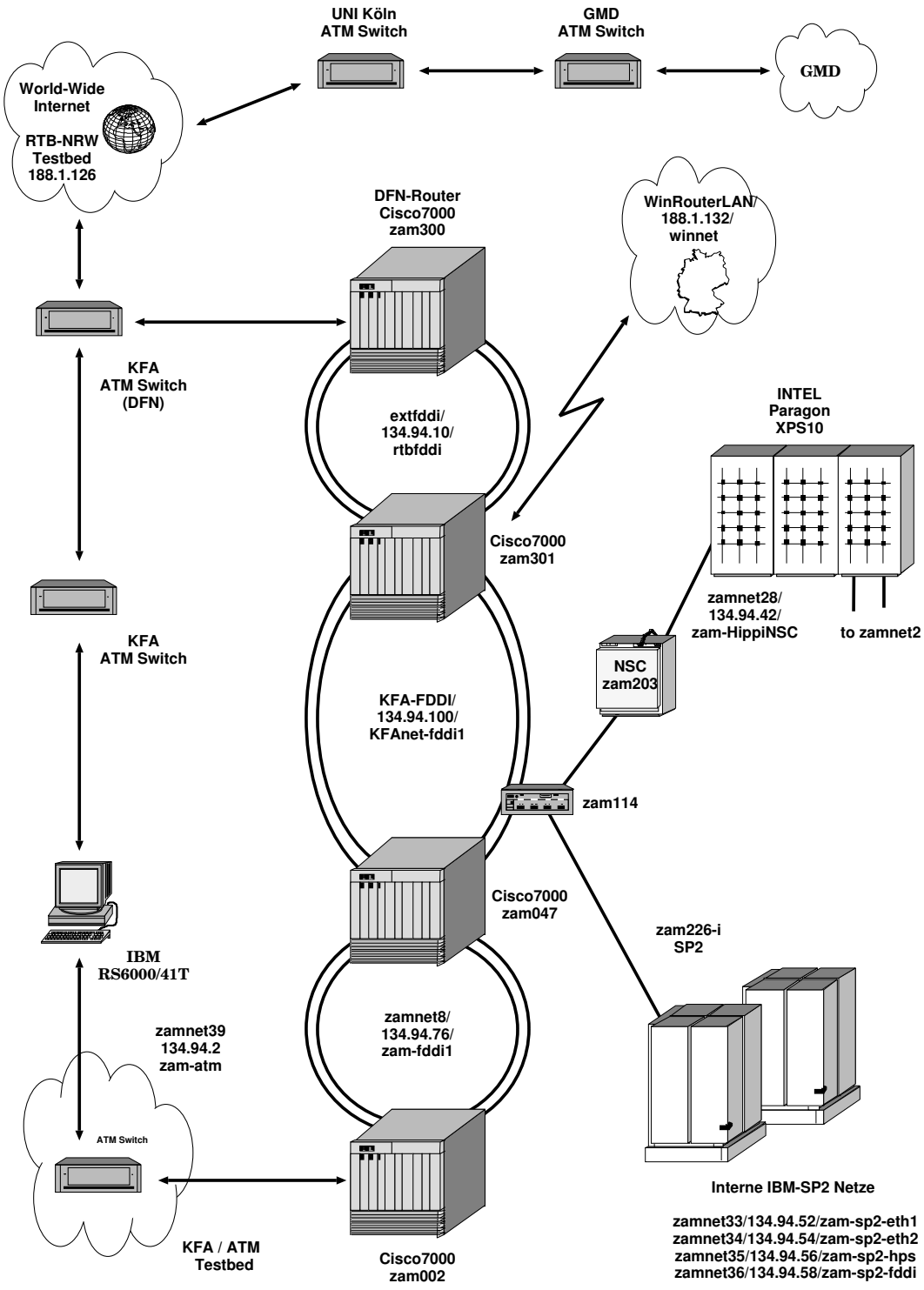

Abbildung 3: Schematische Darstellung der Netzwerktopologie in der KFA Jülich

ist, wurde eine IBM RS/6000 *workstation* mit TAXI-Interface (100 Mbit/s) verwendet, um den optimalen Fall einer durchgängigen ATM-Verbindung zwischen dieser *workstation* und der GMD-SP2 (vgl. Abb. 4) testen zu können. Dabei

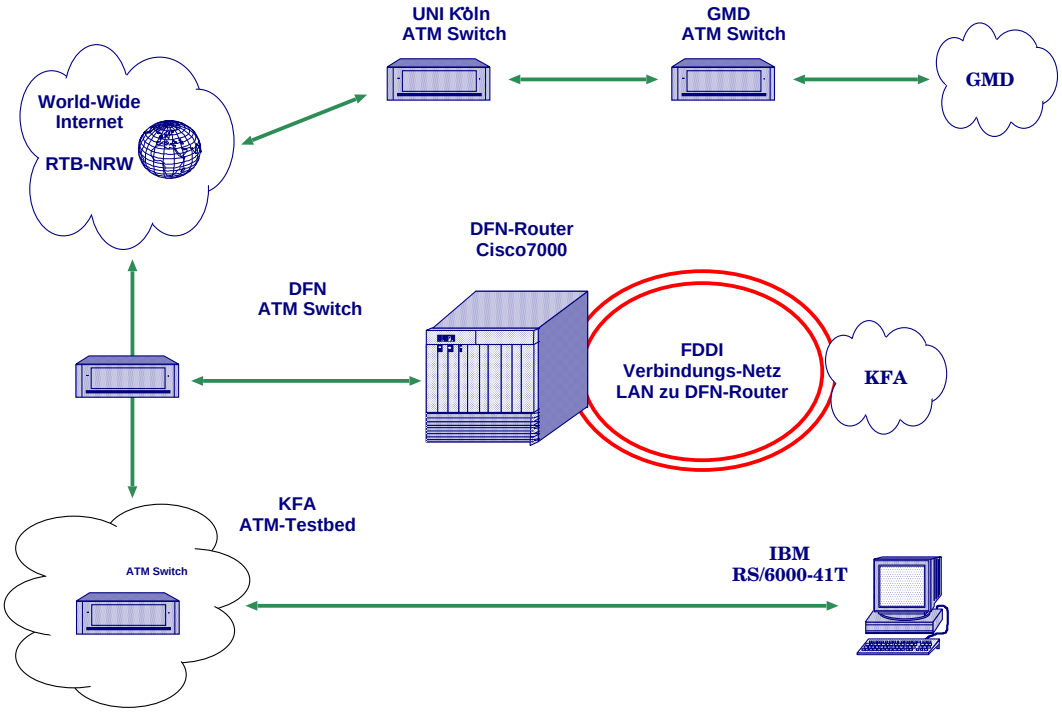

Abbildung 4: Sonderfall einer durchgängigen ATM-Verbindung.

konnte eine Latenz in der Größenordnung von 3.6 ms und eine Bandbreite von 2.7 MByte/s (bzw. 21.6 MBit/s) gemessen werden, wie in Abb. 5 zu sehen ist. Hiermit scheint das mit dem Protokoll TCP/IP physikalisch Machbare erreicht. Verglichen mit der Verbindung XP/S (KFA) – SP2 (GMD) ist dies eine Verbesserung um einen Faktor 2 (Latenz) bzw. mehr als 10 (Bandbreite) (vgl. Abb. 2). In der Produktionsumgebung mit zwischengeschalteten Routern erhält man die Werte 4.5 ms und 1.0 MByte/s.

Es hat sich also gezeigt, daß bei der zur Verfügung gestellten 34 MBit/s Leitung maximal ca. 21.6 MBit/s genutzt werden können. Für den Metacomputer, wie er in diesem Projekt eingesetzt wurde, ergab sich aufgrund von Hardwarebedingten Einschränkungen nur eine Leistung von 1.6 MBit/s.

3 Die benutzten Rechner

Bereits im Herbst 1994 zeigten sich wirtschaftliche und strukturelle Probleme der Firma Thinking Machines Corporation, so daß die GMD aufgrund ihrer an-

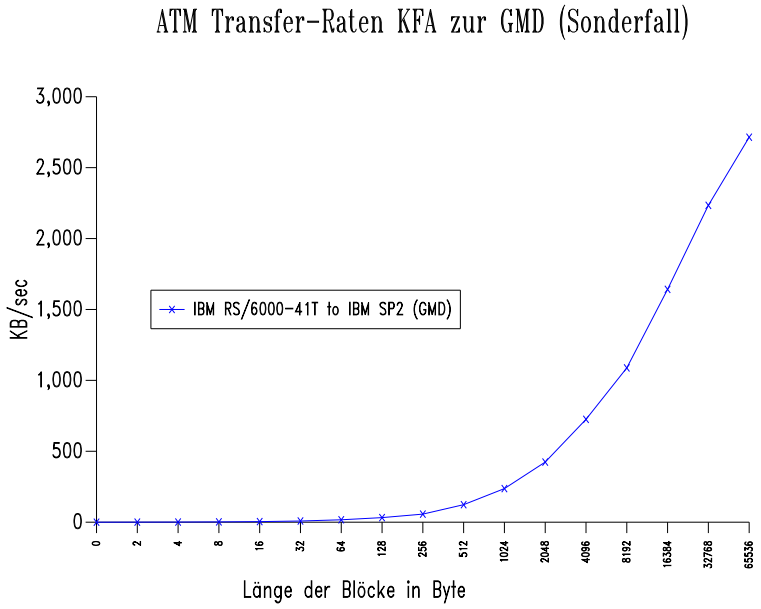

Abbildung 5: Meßergebnisse bei einem ATM-Transfer zwischen der KFA und der GMD

gespannten Haushaltssituation beschloß, den Betrieb der TMC CM-5 Ende 1995 einzustellen. Aus diesem Grund mußte in dem Projekt die CM-5 durch den neuen Rechner IBM SP2 ersetzt werden. Diese Umstellung konnte ohne weitergehende negative Folgen für das Projekt durchgeführt werden, da eine Fortran-Umgebung mit der notwendigen Kommunikations-Bibliothek auch auf der IBM SP2 zur Verfügung stand bzw. portiert werden konnte. Tabelle 2 stellt das System der Intel Paragon in Jülich der IBM SP2 der GMD gegenüber. Obwohl beide Rech-

	XP/S 10	SP2
Anzahl CPUs	140	37
Hauptspeicher pro CPU [MBytes]	32	128
Rechenleistung pro CPU [Mflops]	75	250
Interne Bandbreite [MByte/s]	90	34
Interne Latenz [Mikrosekunden]	38	45
Topologie	2D-Mesh	Crossbar
Netzwerk-Schnittstellen	Ethernet, FDDI/HiPPI	Ethernet, HiPPI, ATM

Tabelle 2: Technische Daten der Intel Paragon XP/S 10 und der IBM SP2

ner zur Klasse der massiv-parallelen Systeme gehören, zeigen ihre technischen Daten sehr unterschiedliche Eigenschaften auf. Während sich die Paragon durch

ihr gutes internes Kommunikationsverhalten, also kleine Latenz und hohe Bandbreite auszeichnet, liegen die Vorteile der SP2 in ihrem großen Speicher und der relativ hohen Rechenleistung des einzelnen Prozessors.

4 Kommunikationsbibliotheken

Um den Metacomputer, der sich jetzt aus dem RTB-NRW und den beiden hier vorgestellten Rechnern besteht, effizient programmieren zu können, mußte eine geeignete Kommunikations-Bibliothek gefunden werden, da eine direkte Programmierung z.B. mittels *sockets* zu aufwendig gewesen wäre. Ein wichtiges Teilziel des Projektes war es, existierende Schnittstellen auf ihre Eignung für Metacomputing-Projekte zu untersuchen. Eine geeignete Bibliothek muß folgende Voraussetzungen erfüllen:

- Die Performance innerhalb der Rechner muß mit der Leistung der jeweiligen rechnerspezifischen Kommunikations-Bibliothek vergleichbar sein.
- Es muß eine ortstransparente, logische Verbindung zwischen den Prozessoren der benutzten Rechner möglich sein. Dies wird im weiteren mit dem Begriff *Konnektivität* bezeichnet. Der Unterschied zwischen einer externen und einer internen Verbindung muß für die Programmierung transparent werden.
- Es muß gewährleistet sein, daß die Entwickler der Bibliothek ausreichend Unterstützung z.B. für eventuelle Fehlerbeseitigung oder Funktionalitätserweiterungen bieten.

Im einzelnen wurden zu Beginn des Projektes folgende Bibliotheken auf Eignung untersucht:

- PVM
  Parallel Virtual Machine (PVM) wurde als Public-Domain-Werkzeug zur Parallelverarbeitung in heterogenen Workstation-Netzen am Oak Ridge National Laboratory entwickelt [4]. Die Portierung auf massiv-parallele Systeme erfolgte erst zu einem späteren Zeitpunkt der Programmentwicklung. Gegenwärtig stellt PVM einen Quasi-Standard dar. Der Hauptvorteil, die Portabilität zwischen den verschiedensten Plattformen, ist auch gleichzeitig der Hauptnachteil, wenn eine hohe Effizienz der Implementierung auf massiv-parallelen Systemen angestrebt wird. Verschiedene Hersteller paralleler Systeme tragen dem Rechnung, indem sie eigene Implementierungen vorgenommen haben. Auch wenn einerseits diese Implementierungen die Effizienz von PVM erhöhen, wird andererseits die Konnektivität eingeschränkt, so daß z.B. eine Kopplung mit anderen Systemen ausgeschlossen ist. Auch schränken spezielle Erweiterungen oder ein alter Release-Stand die Portabilität weiter ein.

- MPI
  Das *Message Passing Interface* (MPI) [8] ist die neueste und fortschrittlichste Schnittstelle: neben dem üblichen Kern von Sende-, Empfangs- und Synchronisationsoperationen sind weitergehende Konzepte wie z.B. die Definition von Prozeßgruppen, Kommunikationskontexte, ausdrucksstarke globale Operationen etc. realisiert. Unter dem Aspekt des Metacomputing ist MPI gerade aufgrund dieser Konzepte der interessanteste der hier betrachteten Ansätze. Allerdings ist die Möglichkeit, unterschiedliche Rechner miteinander zu koppeln, nur in sehr begrenztem Umfang vorhanden: einerseits bieten die sehr effizienten, rechnerspezifischen Implementierungen einzelner Hersteller, wie z.B. Intel oder IBM, dies gar nicht, andererseits verfügen zwar Public-Domain-Implementierungen über eine entsprechende Schnittstelle, sind dann aber nicht in der Lage, gleichzeitig eine gute Performance innerhalb eines Rechners zu gewährleisten [1].

- DFN-RPC
  Zum Zeitpunkt der Evaluierung der Bibliotheken existierte keine Portierung des DFN-RPC für die damals noch im Projekt befindliche CM-5. Außerhalb der DFN-Nutzergemeinde hat diese Schnittstelle keine Bedeutung. Im Sinne maximaler Portabilität und Verbreitung der Projektergebnisse wurde daher vom Einsatz des DFN-RPC abgesehen.

- PARMACS
  Bei den *Parallel Macros* (PARMACS) [2] handelt es sich um eine Entwicklung zur Programmierung von Parallelrechnern mittels Message-Passing, die ursprünglich aus dem Argonne National Laboratory stammt und die in die achtziger Jahre zurückreicht. Während eines Forschungsaufenthaltes in Argonne hat ein GMD-Mitarbeiter die ursprünglich für die Sprache C formulierten Macros nach FORTRAN portiert und anschließend in der GMD (St. Augustin) weiterentwickelt. Eine Reihe von Implementierungen haben, insbesondere durch EU-Projekte zur Verbreitung gerade im europäischen Raum beigetragen (GENESIS-, RAPS-Benchmark). So finden sich auf der Basis der PARMACS ein Reihe umfangreicher Codes, auch mit Produktionscharakter (EU-Projekt EUROPORT [3]).
  In der KFA, insbesondere aber in der GMD, sind die PARMACS auf verschiedenen Systemen im Einsatz, und entsprechende Erfahrungen bei der Programmierung konnten bereits gewonnen werden. Von den PARMACS erhält man kommerzielle Implementierungen (Firma PALLAS GmbH, Brühl). Diese erwiesen sich als stabil und nutzen effizient die verwendete Hardware.

Ein zu Beginn dieses Projektes durchgeführter qualitativer Vergleich der betrachteten Bibliotheken, die alle nicht den gesamten, geforderten Funktionsumfang abdecken, zeigt Tabelle 3. Offensichtlich wiesen alle betrachteten Bibliothe-

	MPI	PARMACS	PVM
Effizienz auf SP2	?[1]	+	+
Effizienz auf XP/S 10	+	+	0
Verbreitung	?	+	+
Support	-	+	-
Konnektivität	-	-	-

Tabelle 3: Qualitativer Vergleich der Message-Passing-Bibliotheken
(+ = gut, 0 = befriedigend, - = schlecht, ? = unbekannt)

ken bezüglich der Konnektivität ein Defizit auf. Ebenso sieht man, daß die PARMACS die geforderten Kriterien am besten abdeckten. Im Herbst 1994 führten die Projektpartner daher Gespräche mit der Firma PALLAS GmbH, Brühl, die Implementierungen der PARMACS entwickelt und vermarktet. PALLAS konnte ein Konzept für die Erweiterung der PARMACS Version 6.1 vorlegen, das den Projektpartnern geeignet schien, die besonderen Anforderungen zur Kopplung von unterschiedlichen Rechnern zu erfüllen, und die Firma PALLAS erhielt den Auftrag, die notwendigen Erweiterungen auszuarbeiten und zu implementieren. Neben der Tatsache, daß die GMD bereits Erfahrung mit dieser Software hatte sammeln können und daß bereits effiziente Implementierungen der PARMACS für die SP2 und die Paragon existierten, stand mit der PALLAS GmbH zudem ein kompetenter Ansprechpartner zur Verfügung.

Die Version PARMACS 7.1 wurde im August 1995 für die Intel Paragon und im November 1995 für die IBM SP2 ausgeliefert. Es waren allerdings erwartungsgemäß mehrere Iterationen von Nachbesserungen durch PALLAS notwendig, bis die gewünschte Effizienz und Stabilität auf den beiden Zielsystemen erreicht war. Hier machte es sich erfreulicherweise bemerkbar, daß die Firma PALLAS ausreichend Unterstützung bieten konnte und auf Fehlerreports schnell reagierte. Die aktuellen Meßergebnisse sind in Tabelle 4 dargestellt, Abb. 6 zeigt exemplarisch den direkten Vergleich auf der Intel Paragon. Während die Meßergebnisse für

	Intern				Extern	
	Intel XP/S 10		IBM SP2		XP/S 10 ↔ SP2	
	NX	PM7	MPL	PM7	*sockets*	PM7
Latenz [ms]	0.038	0.065	0.045	0.107	5.0	30.0
Bandbreite [MByte/s]	86	84	35	34	0.450	0.250

Tabelle 4: Message-Passing: Interne und externe Kommunikationsleistungen

[1]Mittlerweile existiert auch für die IBM SP2 eine von Hersteller gewartete MPI-Bibliothek, die eine gute Performance bietet, zur Zeit des Auswahl war dies allerdings noch nicht der Fall. Auch hat sich mittlerweile eine weite Verbreitung von MPI ergeben. Noch ungelöst ist aber zur Zeit das Problem der Konnektivität.

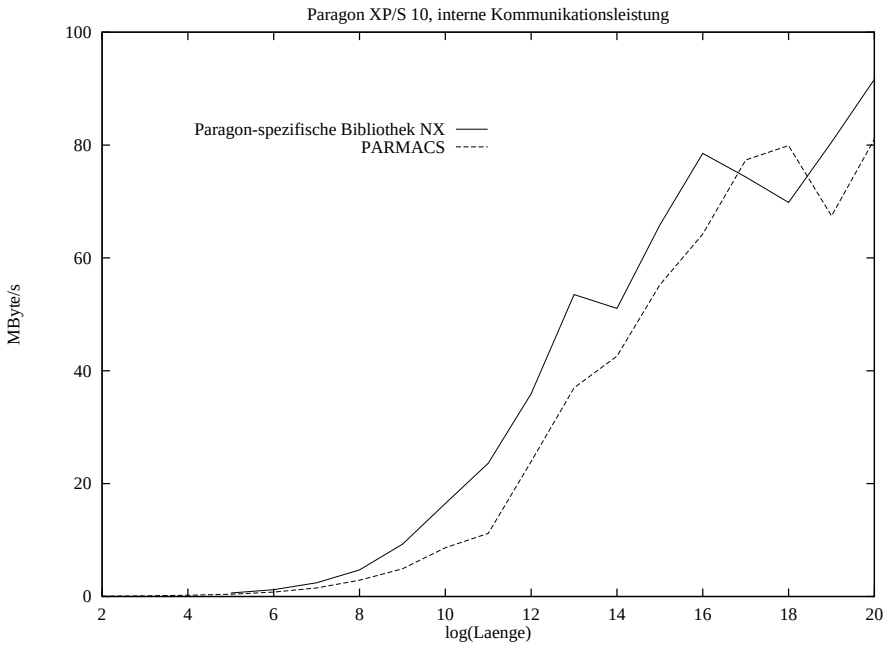

Abbildung 6: Vergleich der interne Nachrichtenleistung
auf der Intel Paragon XP/S 10

die interne Kommunikation zufriedenstellend sind, gibt es doch eine erhebliche Einbuße in der Latenz bei der verteilten Anwendung. Eine Ursache dafür ist die Datenkonvertierung, die zwischen den beiden Systemen notwendig ist, obwohl sich beide Systeme an den IEEE-Standard halten. Die Intel Paragon und die IBM SP2 benutzen ein unterschiedliches Datenformat (*Little-Endian* auf der SP2 bzw. *Big-Endian* auf der Paragon). Um also einen Datensatz von einem Rechner zum anderen zu schicken, müssen die Daten von dem einen Format in das andere Format übersetzt werden. Bei der Kommunikation mittels Unix-*sockets* wurden die Daten nicht konvertiert, so daß sich allein dadurch eine niedrigere Latenz und größere Bandbreite erklärt. Mit Hilfe dieser Bibliothek konnte nun das Anwendungsprogramm TRACE parallelisiert und auf die beiden Rechner portiert werden.

5 TRACE

Das Programm TRACE (Transport of Contaminants in Environmental systems) [12] ist eine Anwendung aus dem Bereich der Umweltwissenschaften. Es berechnet den Wasserfluß in porösen heterogenen Medien und bildet somit die Grundlage für die Simulation des Stofftransportes im Boden und Grundwasser anhand eines dreidimensionalen Modells, vgl. Abb. 7. Unter der Leitung von Dr. H. Vereecken wird TRACE im Institut für Chemie und Dynamik der Geosphäre (ICG-4) der KFA bezüglich der verwendeten Methoden und Algorithmen permanent weiter-

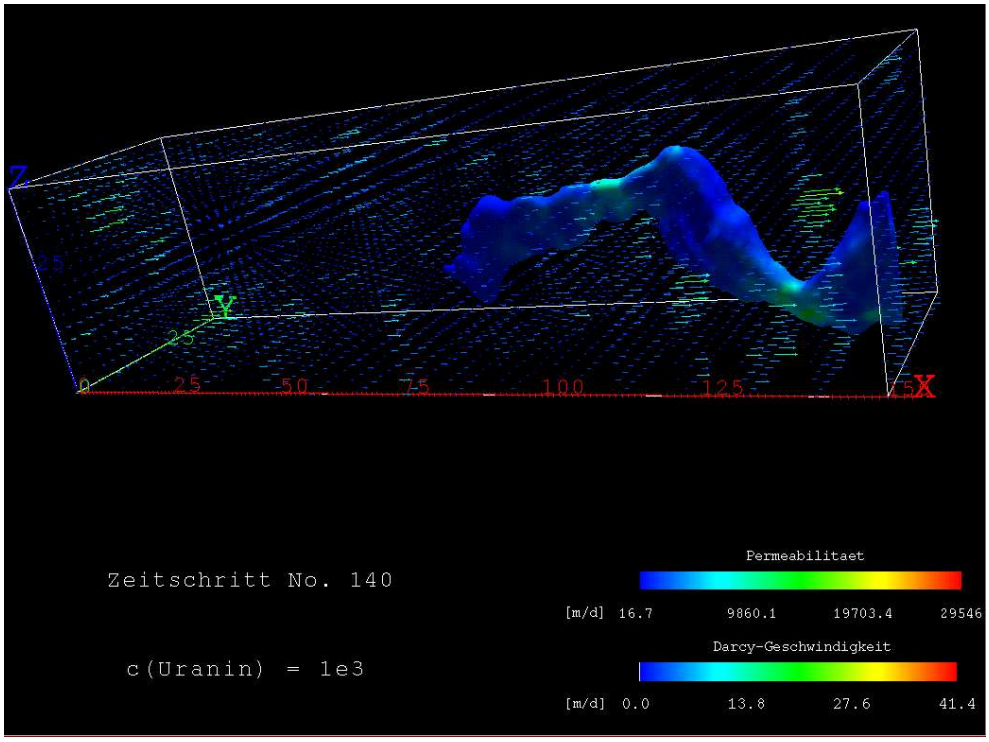

Abbildung 7: Ausbreitung einer Schadstoffwolke im Boden als Ergebnis des Programms TRACE.

entwickelt. Ursprünglich wurde der Code für eine CRAY X-MP in FORTRAN 77 geschrieben und verwendet eine Diskretisierung von Ort und Zeit durch Finite-Elemente (FE) und Finite-Differenzen. Der Speicherplatz auf der CRAY ließ allerdings nur die Berechnung von 5×10^4 FE-Knoten zu. Da sinnvolle Gebietsgrößen erst durch rund 10^6 FE-Knoten beschrieben werden, wurde bereits früh über eine Parallelisierung des Programms nachgedacht.

Gemeinsam mit Mitarbeitern aus dem ZAM wurde eine parallele Version von TRACE für die Intel Paragon XP/S 10 erstellt. TRACE wurde hierbei unter Verwendung einer Gebietszerlegung und des Schwarzschen Verfahrens erweitert. Hierbei wurde insbesondere auf eine Reduzierung der notwendigen Kommunikation geachtet [13], da gerade die langsamen Netzverbindungen in einem verteilten Rechner einen deutlichen Einfluß auf das Gesamtergebnis haben. Das parallelisierte Programm zeigt eine sehr homogene Struktur: jeder Prozessor bekommt die gleiche Gebietsgröße zugewiesen. Dadurch führt jeder Prozessor ungefähr die gleiche Arbeit aus; es liegt eine gute Lastbalancierung vor.

Bei der Entwicklung des Programms wurden die Intel-spezifischen NX-Befehle [5] zur Kommunikation zwischen den Rechenknoten in Unterprogramme gekapselt, so daß durch geeignete Modularisierung die Zahl dieser notwendigen Schnittstellen gering (≤ 10) gehalten werden konnte. Die später notwendige Portierung mit Hilfe der Kommunikations-Bibliothek PARMACS wurde deutlich vereinfacht.

Hier zahlte sich schon das Vorhandensein einer portablen, effizienten Kommunikationsbibliothek aus, da nur eine Version von TRACE erstellt und gewartet werden mußte.

6 Hilfsprogramme

Projektbegleitend wurden an der KFA weitere Arbeiten durchgeführt, die einerseits der Visualisierung der numerischen Ergebnisse, andererseits der Programmanalyse und -optimierung dienten. Das Kommunikationsverhalten und die Interpretation der Ergebnisse in einem größeren Programm sind derart komplex, daß graphische Hilfsmittel hierfür unabdingbar sind.

So ist die Visualisierung der numerischen Ergebnisse und der Gebietszerlegung extrem wichtig. Bei der großen Menge an Ergebnisdaten (mehrere MByte pro Lauf) kann anhand der reinen Daten ein Programmfehler nicht mehr festgestellt werden. Auch ist eine Interpretation und Analyse der Ergebnisse aufgrund der Zahlenergebnisse praktisch nicht mehr möglich. Um dies benutzerfreundlich zu gestalten, wurde eine Umgebung unter der Software AVS [6] erstellt, die neben der Darstellung der numerischen Ergebnisse, wie in Abb. 7, auch die vorgenommene Gebietszerlegung darstellt. Es sind verschiedene Anzeigemodi nutzbar. In Abb. 8 ist exemplarisch eine Aufteilung des Gebietes auf zwölf Prozessoren zu sehen, dargestellt durch die einzelnen Würfel. Die numerischen Ergebnisse, im Beispiel die Stärke des Wasserflusses, ist durch die verschiedenen Farben gekennzeichnet.

Die Fehlersuche, Optimierung und auch die Lastverteilung werden durch die im ZAM entwickelte X Window basierte Visualisierungsumgebung VAMPIR unterstützt [9]. Mit VAMPIR kann das Laufzeitverhalten eines Programms aufgezeichnet und anschließend angezeigt werden. Neben dem Beginn und dem Ende der einzelnen Prozeduren werden auch Nachrichten, die zwischen den einzelnen Prozessoren gesendet werden, angezeigt. Dazu gehören auch weitere Informationen wie die jeweilige Nachrichtenlänge und -Kennung und die gemessene Bandbreite. Entsprechende Statistiken sind für frei wählbare Teile eines Programmlaufs abrufbar. Eine ausführliche Darstellung der umfangreichen Möglichkeiten von VAMPIR findet man in [9]. Auch für die Auswertung der in diesem Projekt erzielten Meßergebnisse wurde dieses Werkzeug benutzt.

Es hat sich gezeigt, daß die besonderen Eigenheiten eines heterogenen System Änderungen an Vampir notwendig machten. Dies betrifft z.B. die notwendige Synchronisation der Uhren in einem verteilten System, eine klarere Trennung der unterschiedlichen Rechner bei der Anzeige, Berücksichtigung der unterschiedlichen Taktrate der Rechner etc. Die Änderungen wurden projektbegleitend in der KFA durchgeführt.

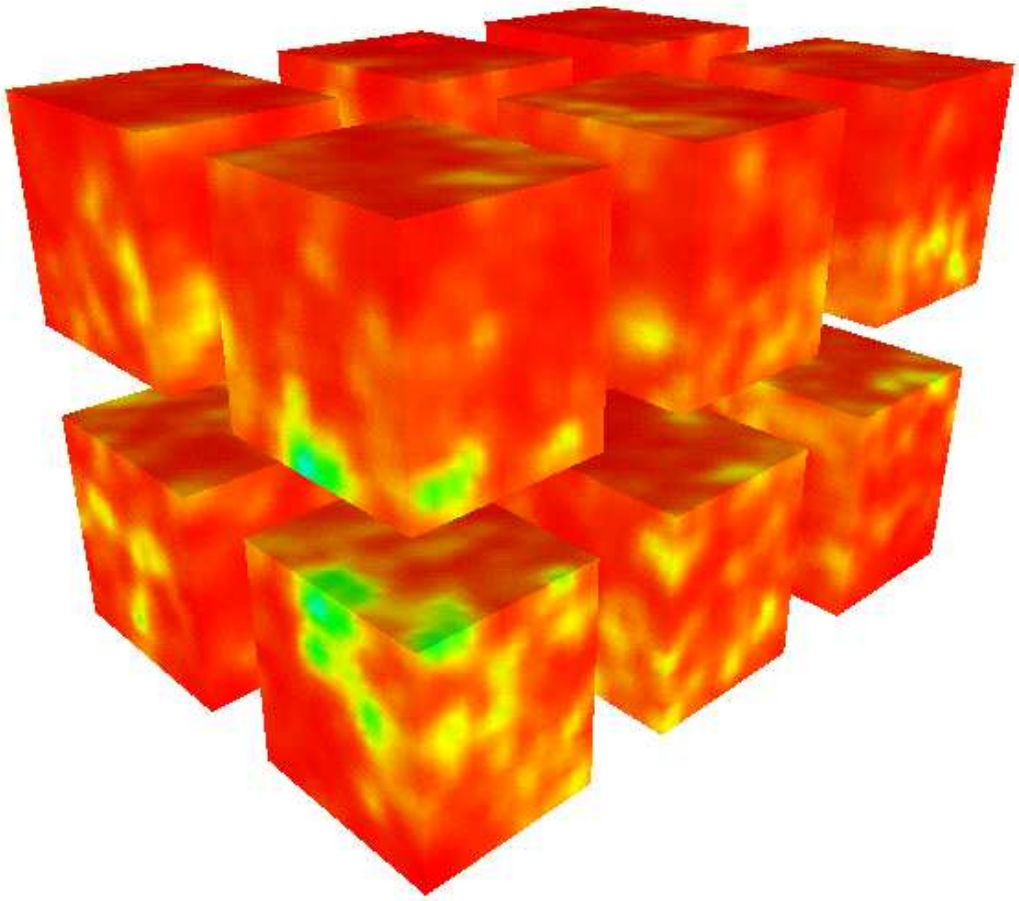

Abbildung 8: Darstellung der numerischen Ergebnisse
bei einer Gebietszerlegung auf 12 Gebiete.

7 Ergebnisse

Es wurde eine Vielzahl von Messungen und Experimenten unternommen, um das Verhalten des Programms sowohl auf einem einzelnen Rechner als auch auf dem verteilten System zu untersuchen. Es kann hier nur ein kleiner Ausschnitt dieser Ergebnisse präsentiert werden. Tabelle 5 faßt die wichtigsten Ergebnisse zusammen. Es wurde ein TRACE-Lauf mit einem Beispieldatensatz durchgeführt, wobei eine Diskretisierung mit insgesamt 31x31x31 Knoten zugrunde lag. Mit dieser Konfiguration wurden 15 Zeitschritte gerechnet. Die Anzahl von 15 Zeitschritten ist zwar relativ gering, jedoch zeigt das Programm auch hier schon sein typisches Verhalten, so daß die Ergebnisse auch für reale, längere Rechnungen gültig sind. Bei den Angaben in Tabelle 5 wurde die Zeit für die initiale Datenverteilung, die zwischen 60 und 160 Sekunden liegt, nicht mit berücksichtigt. Auch wurde eine vorläufige, teiloptimierte Version von TRACE benutzt, die noch nicht auf die

speziellen Gegebenheiten des verteilten, heterogenen Systems angepaßt war. Alle

	Prozessoranzahl	Laufzeit [s]
SP2	4	225.5
	8	152.0
XPS	4	1106.2
	8	734.5
XPS / SP2	2+2	1173.1
	4+4	1397.9

Tabelle 5: Laufzeiten (Mittelwerte) des verteilten Programms für 15 Zeitschritte auf 4 und 8 CPUs

Messungen mußten im Produktionsbetrieb durchgeführt werden und unterlagen entsprechend hohen Schwankungen. So schwankte z.B. die Zeit für die verteilte Rechnung bei 4+4 Prozessoren zwischen 1360 und 1583 Sekunden.

Die Ergebnisse für die Intel Paragon und die IBM SP2 allein skalieren recht gut: durch die Verdopplung der Prozessoranzahl wurde eine Reduktion der Rechenzeit um ca. 30% erreicht. Allerdings sind die Ergebnisse für die verteilten Läufe überraschend: trotz Nutzung der im Vergleich zur Paragon schnelleren SP2-Prozessoren ergab sich keine Reduzierung der Laufzeit. Eine nähere Untersuchung dieses Phänomens wurde mit dem Visualisierungs-Tool VAMPIR vorgenommen. Abbildung 9 zeigt schematisch einen Iterationsschritt des Programms. In rot dargestellt sind die Kommunikationsroutinen, in grün die eigentlichen Berechnungen. Schwarze Linien zwischen den Prozessoren symbolisieren die zwischen den Rechnern ausgetauschten Nachrichten. Es ist deutlich zu sehen, daß die Prozessoren der Paragon (die oberen vier Balken) noch rechnen, während alle Prozesse auf der SP2 nach kurzer Zeit die Rechnungen beendet haben und kommunikationsbereit sind. Die zusätzliche Rechenkapazität der SP2 wird somit nicht genutzt, sondern sie wird durch untätiges Warten vergeudet. Ursache für dieses Ungleichgewicht ist, daß die auszuführende Rechenarbeit gleichmäßig auf die Prozessoren verteilt wurde und nicht gemäß der tatsächlichen Rechenkapazität der Prozessoren. Durch Verwendung eines heterogenen Rechnersystems entstand ein Lastungleichgewicht, das bei dem homogenen System nicht vorlag. Die im Vergleich zu der Paragon-Version zusätzlichen 70 Sekunden Rechenzeit sind durch die langsame externe Kommunikation und die Notwendigkeit der Datenkonvertierung zu erklären.

Die Ursache für den Anstieg der Rechenzeit bei Hinzunahme weiterer Prozessoren sind die steigenden Kommunikationskosten, die sich durch die langsame externe Verbindung ergeben. Diese langen Laufzeiten der Nachrichten werden besonders bei der initialen Datenverteilung gut sichtbar, siehe Abb. 10. Da die Problemgröße für alle Läufe des Programms identisch ist, steigt mit der Anzahl der Prozessoren auch der Kommunikationsanteil; der anteilige Rechenaufwand

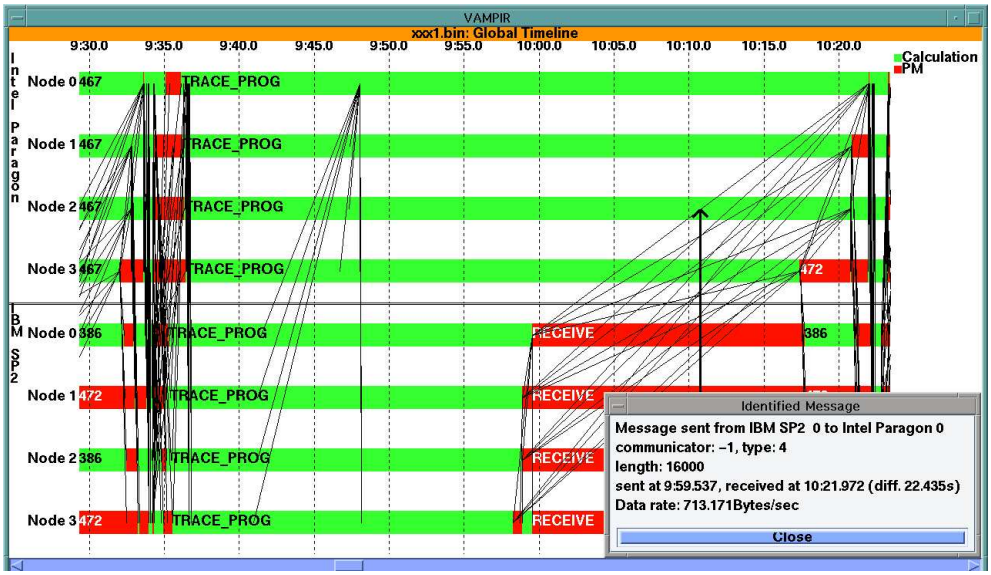

Abbildung 9: Die schematische Darstellung einer Iteration des Programmes TRACE verdeutlicht die ungleiche Lastverteilung

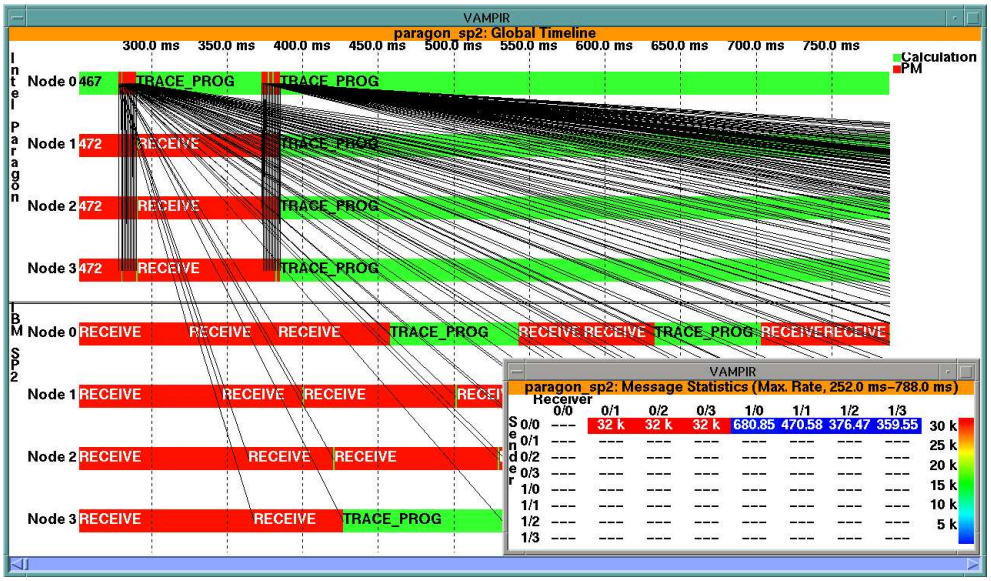

Abbildung 10: Unterschiedliche Nachrichtenlaufzeiten bei der initialen Datenverteilung

pro Prozessor sinkt. Somit verhindert die im Verhältnis zur internen Kommunikation langsame externe Kommunikation, daß durch die Hinzunahme weiterer Prozessoren in diesem Fallbeispiel eine Reduzierung der Rechenzeit erreicht werden kann.

Um also TRACE effizient auf dem Metacomputer rechnen zu lassen, muß die

interne Programmstruktur geändert werden. Dies betrifft zuallererst den Lastausgleich, so daß die Rechenkapazität der SP2 auch tatsächlich genutzt werden kann.

Als Alternative zu TRACE wurde das Programm BD vom Institut für Festkörperforschung der KFA untersucht [14]. BD berechnet die Greensche Funktion für periodische Kristalle. Die Parallelisierung des Codes erfolgte zuerst für die Intel Paragon mit der NX-Kommunikations-Bibliothek. Dabei werden die Werte der Greenschen Funktion für alle 16 betrachteten Elektronenniveaus parallel berechnet, bevor in einer abschließenden globalen Operation der eigentliche Wert bestimmt wird. Dieses Vorgehen wird mehrfach iteriert, um das richtige Funktionsergebnis zu erhalten. Charakteristisch für den Code ist, daß das letzte Elektronenniveau mit einer größeren Genauigkeit berechnet werden muß. Dies bedeutet, daß ein Knoten einen deutlich höheren Rechenaufwand hat als die anderen. Dieses Verhalten ist am Beispiel von zwei Iteration in Abb. 11 zu sehen. Während sich alle Knoten bereits in der rot dargestellten, globalen Operation be-

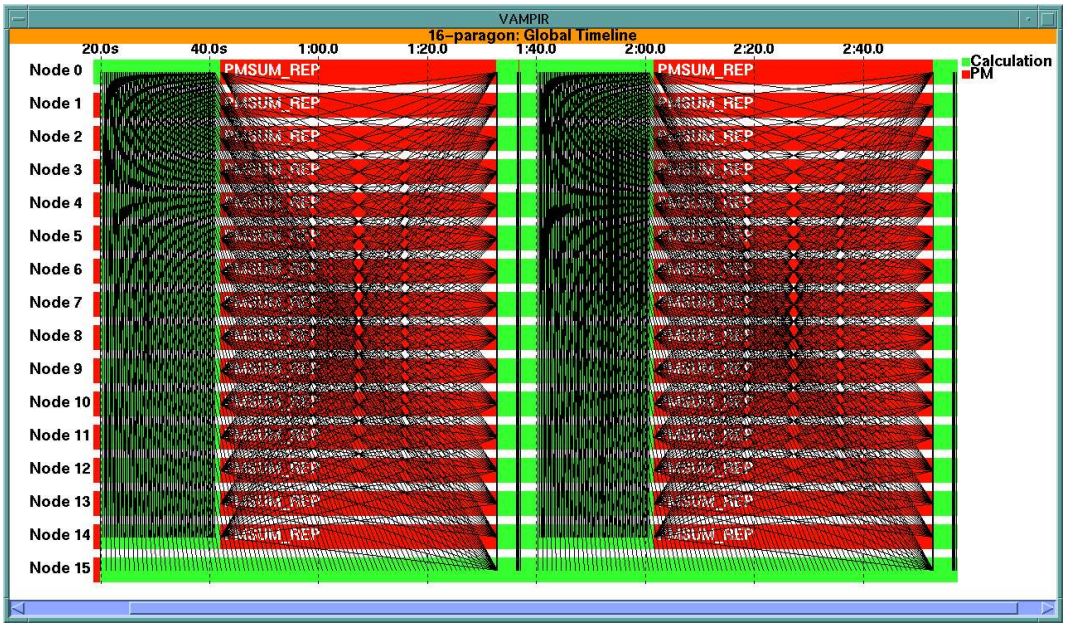

Abbildung 11: Heterogenes Verhalten der Anwendung BD.

finden, hat der letzte Knoten seine grün markierten, lokalen Berechnungen noch nicht beendet, sondern muß noch ca. 50 Sekunden weiterrechnen. In dieser Zeit sind alle anderen Prozessoren untätig, ihre Rechenkapazität ist verloren. Einen Gesamtüberblick über die Zeit, die die jeweiligen Prozessoren für die Kommunikation und für die eigentliche Berechnung brauchen, ist in Abb. 12 zu sehen. Dieses Programm mit seiner heterogenen Struktur ist nun offensichtlich ein besonders geeigneter Fall, um eine heterogene Rechnerstruktur in Form eines Metacompu-

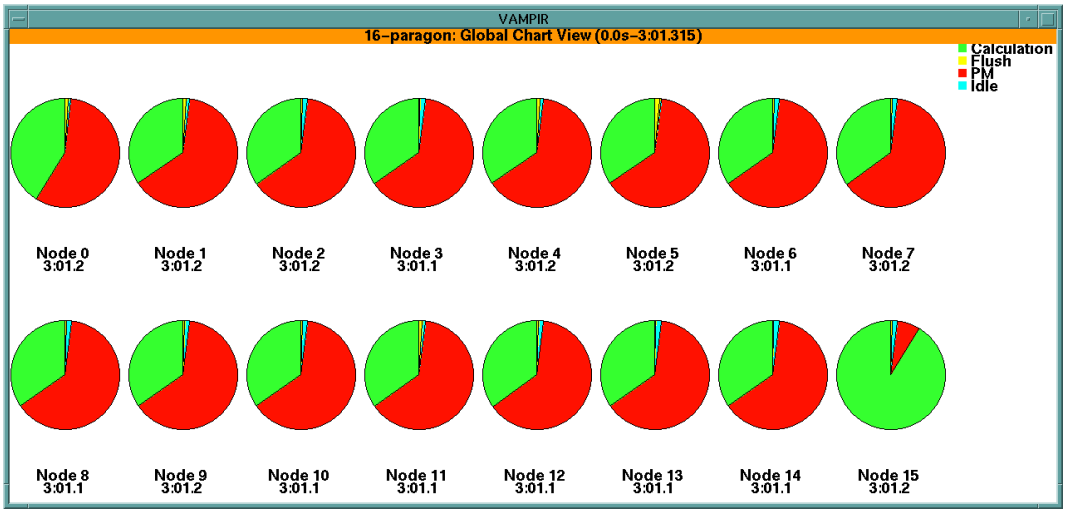

Abbildung 12: Aufteilung der Gesamtzeit auf Rechnung und Kommunikation für die Anwendung BD auf der Paragon

ters zur Bearbeitung einzusetzen.

Dazu wurde zunächst, analog zu TRACE, mittels PARMACS eine Portierung von BD auf die SP2 vorgenommen. Anschließend wurde das Programm ohne weitere Änderung auf dem Metacomputer, bestehend aus 15 Paragon-Knoten und einem SP2-Knoten, bearbeitet, wobei der letzte, rechenintensive Prozeß auf der SP2 gestartet wurde. Bei 10 Iterationen ergab sich eine Reduktion der Rechenzeit von 882 auf 476 Sekunden (bzw. bei dem in den Abbildungen gewählten Ausschnitt eine Reduktion von 181 Sekunden auf 103 Sekunden. Abb. 13 zeigt das Verhalten der Anwendung auf dem Metacomputer. Es ist sehr gut zu sehen, daß auf dem Metacomputer viel weniger Zeit für die Kommunikation aufgewendet werden muß, da jetzt keine Zeit mehr dadurch verloren geht, daß auf den einzelnen, länger rechnenden Prozessor gewartet werden muß. Auch der Überblick über die Verteilung der Gesamtzeit in Abb. 14 zeigt dies deutlich.

8 Ausblick

Unabdingbar für den Erfolg einer Metacomputing-Anwendung ist eine sorgfältig ausgewählte Kommunikations-Bibliothek. Sie muß ueber eine ausreichende Kommunikationsleistung innerhalb der einzelnen Rechner verfügen, aber auch eine eine schnelle Verbindung mit anderen Rechnern ermöglichen, was eine breitbandige Netzverbindung zwischen den beteiligten Rechnern voraussetzt. Wenn sie darüber hinausgehend sowohl für die interne als auch für die externe Kommunikation die gleichen Kommunikationsprimitiva zur Verfügung stellen, ist eine parallele Anwendung mit moderatem Aufwand zu einer Metacomputing-Anwendung zu machen. Die wichtigste Erkenntnis ist, daß die benutzten Rechner ein einheitli-

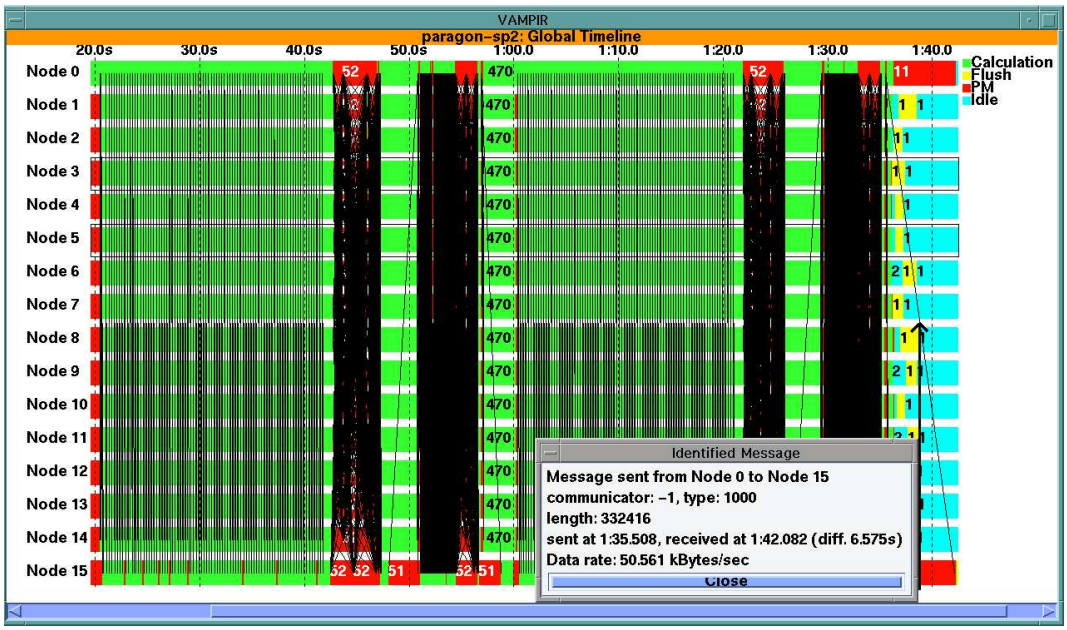

Abbildung 13: Anwendung BD auf dem Metacomputer.

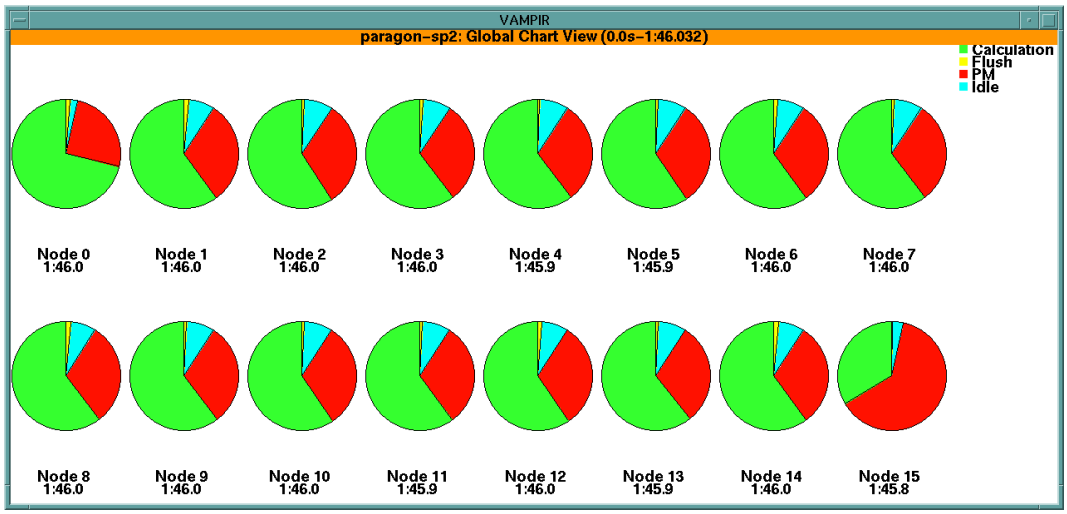

Abbildung 14: Aufteilung der Gesamtzeit auf Rechnung und Kommunikation für die Anwendung BD auf dem Metacomputer

ches Datenformat haben müssen, um Datenkonvertierung und damit erheblichen zusätzlichen Aufwand vermeiden.

Dies allein reicht aber nicht aus, wenn durch die gegebenenfalls notwendige Datenkonvertierung wieder Kapazität verloren geht. Um einen Metacomputer effizient betreiben zu können, müssen die beteiligten Systeme über ein einheitliches Datenformat verfügen, so daß dieser zusätzliche Aufwand nicht nötig ist. Es reicht dazu allerdings nicht aus, einen Standard wie z.B. IEEE zu nutzen, wenn

die interne Repräsentation unterschiedlich ist (*Little Endian* und *Big Endian*). Hier bieten neuere, einheitliche Systeme deutliche Vorteile.

Die effiziente Nutzung eines Metacomputer erfordert eine aufwendigere Parallelisierung des Programms. Gerade Programme, die eine homogene Struktur aufweisen, wie das hier benutzte Programm TRACE, verlieren zunächst an Effizienz, wenn sie von einem Metacomputer bearbeitet werden. Zuerst muß für eine entsprechende Lastbalancierung gesorgt werden, damit die zusätzliche Rechenkapazität des schnelleren Rechners nicht durch unnötige Wartezeiten verloren geht. Dazu ist eine Änderung des bestehenden Programms notwendig. Weiterhin ist es wichtig, daß die langsame externe Kommunikation mit Berechnungen überlagert wird, damit nicht allein dadurch ein Effizienzverlust eintritt.

Die Portierung auf ein heterogenes System ist hingegen einfacher, wenn das zugrundeliegende Programm selbst über eine heterogene Struktur verfügt, wenn also die einzelnen Prozesse über einen unterschiedlichen Rechenbedarf verfügen. In diesem Fall kann ein Metacomputer ohne weitere Programmänderungen effizient genutzt werden, indem die jeweiligen Prozesse auf den entsprechenden schnelleren oder langsameren Rechner bearbeitet werden. Dieser Effekt wird natürlich signifikanter, wenn nicht - wie hier im Projekt geschehen - gleichartige Systeme, sondern Rechner mit unterschiedlichen Architekturen, also z.B. Vektorrechner und Parallelrechner eingesetzt werden.

Eine Vision für die Zukunft ist es, diese Vorgänge zu automatisieren und nicht nur einzelne Prozesse fest von einem Prozessor bearbeiten zu lassen. So ist es denkbar, daß die einzelne Prozesse zur Laufzeit migrieren, um auf einer für sie optimalen Rechnerarchitektur bearbeitet zu werden und gleichzeitig die Gesamtlaufzeit zu minimieren. Leider sind wir davon noch weit entfernt: die Schwierigkeiten beginnen mit dem Bestimmen der geeigneten Architektur für einen einzelnen Programmteil, bevor dieser ausgeführt wird und reichen bis zur Prozeßmigration in einem heterogenen System. Auch ist es fraglich, ob der Aufwand der Migration nicht den Gewinn durch Nutzung der optimalen Architektur zunichte macht.

Erste Schritte auf diesem Weg sind gegangen, aber große Erfolge haben sich bisher, auch aufgrund der gewählten Hardware-Plattformen und der damit zwingend notwendigen Datenkonvertierung, noch nicht in ausreichendem Maße eingestellt. Aber gerade das zugrundeliegende Potential, nämlich die Lösung größerer und komplexerer Probleme, als es jemals möglich war, machen die Forschung in diesem Bereich wichtig. Durch dieses Projekt ist hoffentlich ein Anfang gemacht worden, aufbauend auf dem weitere Erkenntnisse über den Einsatz heterogener Systeme gewonnen werden können.

Danksagung: Wir danken dem BMBF für die Förderung des Projektes sowie dem DFN-Verein für seine Unterstützung.

Literatur

[1] P. Bridges, N. Doss, W. Gripp, E. Karrels, E. Lusk, A. Skjellum: *Users' Guide to MPICH, a Portable Implementation of MPI.* http://www.mcs.anl.gov/mpi/, 1995.

[2] R. Calkin, R. Hempel, H. C. Hoppe, P. Wypior: *Portable Programming with the PARMACS Message-Passing Library.* Parallel Computing, Vol. 20, No. 4, pp. 615-632, 1994.

[3] *Europort - Industrial High-Performance Computing.* In: J. Elliot, K.Stueben (editors): *Proceedings HPCN'95.* Lecture Notes in Computer Science 919, Springer, Berlin, 1995.

[4] A. Geist, A. Beguelin, J. Dongarra, W. Jiang, R. Mancheck, V. Sunderam: *PVM: Parallel Virtual Machine. A Users' Guide and Tutorial for Networked Parallel Computing.* MIT Press, Cambridge, MA., 1994.

[5] Intel Supercomputer Systems Division: *Paragon User's Guide.* No. 312489-002, 1993.

[6] A. Kempkes: *Visualisierung verteilt berechneter, mehrdimensionaler Datenfelder.* Jül-3216, Forschungszentrum Jülich, 1996.

[7] O. A. McBryan: *HPCC: The Interrelationship of Computing and Communication.* In: E. D. Hollander, G. R. Joubert, F. J. Peters, D. Trystran (editors): *PARALLEL COMPUTING: State-of-the-Art and Perspectives.* Elsevier Science B.V., pp. 31-55, 1996.

[8] Message Passing Interface Forum: *MPI: A Message-Passing Interface Standard.* http://www.mcs.anl.gov/mpi/index.html, 1995.

[9] W. E. Nagel, A. Arnold, M. Weber, H.-C. Hoppe, K. Solchenbach: *VAMPIR: Visualization and Analysis of MPI Resources.* Supercomputer 63, Volume XII, Number 1, pp. 69-80, 1996.

[10] R. Niederberger: *Schnelle Netze für das Metacomputing. Anspruch und Wirklichkeit im RTB-NRW.* Interner Bericht, zur Veröffentlichung vorgesehen, 1996.

[11] L. Smarr, C. E. Catlett: *Metacomputing.* Communications of the ACM, Vol. 35, No. 6, pp. 45-52, 1992.

[12] H. Vereecken et al., *TRACE: A Mathematical Model for Reactive Transport in 3D Variably Saturated Porous Media.* KFA/ICG-4 Internal Report No. 501494, 1994.

[13] R. Wimmershoff: *Entwicklung und Implementierung einer dreidimensionalen Partitionierungsstrategie für das Programm TRACE auf einem massivparallelen Rechner.* Jül-3157, Forschungszentrum Jülich, 1995.

[14] R. Zeller: *Green-Function Method for Electronic Structure of Periodic Crystals.* International Journal of Modern Physics C, Vol. 4, No. 6, pp. 51-58, 1993.